N89-26600

A RAPID PROTOTYPING/ARTIFICIAL INTELLIGENCE APPROACH TO SPACE STATION-ERA INFORMATION MANAGEMENT AND ACCESS

Richard S. Carnahan, Jr.
Stephen M. Corey
John B. Snow

Martin Marietta Information & Communications Systems
Denver, Colorado

1. ABSTRACT

This effort has focused, and continues to focus, on applying rapid prototyping and Artificial Intelligence techniques to problems associated with Space Station-era information management systems. In particular, our work is centered on issues related to (1) intelligent man-machine interfaces applied to scientific data user support and (2) the requirement that intelligent information management systems (IIMS) be able to efficiently process metadata updates concerning types of data handled. The advanced IIMS represents functional capabilities driven almost entirely by the needs of potential users. The amount and complexity of scientific data projected to be generated by Space Station-era projects (e.g., Eos) is likely to be significantly greater than data currently processed and analyzed. Information about scientific data must be presented to potential users in a clear and concise way, with support features being incorporated to allow users at all levels of expertise efficient and cost-effective data access. Additionally, since data modifications and additions will frequently occur, mechanisms for allowing more efficient IIMS metadata update processes must be addressed. To this end, our work has examined the following aspects of IIMS design: (1) IIMS data and metadata modeling, including the automatic updating of IIMS-contained metadata, (2) IIMS user-system interface considerations, including significant problems associated with remote access, user profiles, and on-line tutorial capabilities, and (3) development of an IIMS query and browse facility, including the capability to deal with spatial information. A working prototype has been developed and is being enhanced. Future work will attempt to clarify a number of issues which have emerged from our present efforts, particularly concerning IIMS information base structure and its relationship to user-support and distributed, heterogeneous database access.

2. INTRODUCTION

This paper represents the culmination of our early examination of the areas of advanced information management and access using rapid prototyping and 'Artificial Intelligence (AI) programming techniques. In one sense, it is inaccurate to have only one section with the heading 'Results'. The entire paper reflects our current thinking concerning issues surrounding future IIMS operations and development, and all of what is written here is a direct result of our early prototyping efforts. To date, our work has served as a validation of the approach we have used since the initial prototype has helped to crystalize our thinking and drive out more specific system requirements. As a result of what we have accomplished, we are now in a position to move forward. Section 3 details our interpretation of global requirements advanced information systems will necessitate, and our understanding of current system limitations that compel the use of advanced concepts in both system design and implementation. Section 4 discusses our approach to implementing a prototype IIMS using the application domain of large science databases. Section 5 presents the current state of our prototype and concluding remarks concerning implications of our work.

3. BACKGROUND AND PROBLEM

3.1. Current State of Space System Data Management

For decades, NASA and other agencies (e.g., NOAA, NCAR, EROS) have collected vast amounts of scientific data in support of Earth and space research. For the value of this data to be fully realized, it must be easily accessed in a timely manner not only by those researchers directly associated with a particular mission or project, but also by researchers who are involved in the domain of interest as well as those in related scientific fields. To complicate the situation, the amount of data being collected or generated is increasing exponentially and should continue to grow in the future [15]. The end result is that many researchers cannot use data they need because they do not *know how* to access the data, they do not *have* access to the data, or they do not *know where or if* the data exists.

Currently, data (we use the term 'data' to refer to all collected and generated data as well as information about data [*metadata*]) reside with the agency sponsoring the data collection effort or, in some cases, in large repositories. Some of these facilities publish catalogs from which the user can order data, but they usually provide no means for remote access. Such a policy has many obvious drawbacks; the most serious is difficulty in locating and obtaining all suitable data in a short period of time. If users are fortunate enough to find needed data, they must often convert the data into a form their algorithms can process before it can be studied. Even facilities or projects having computerized data access have many limitations when it comes to locating all relevant information, and usually lack sophisticated querying and user-system interface techniques. These systems tend to

be built in such a manner that makes access very difficult by someone unfamiliar with the data. With the projected increase in amounts and complexity of acquired data, the need to have data available to all researchers, not just those directly involved, and the increasing need to understand relationships among data, it is rapidly becoming impossible to access the most suitable information.

3.2. New Directions

NASA is progressing with new information gathering programs that will produce significant amounts of new information in the 1990s (e.g., EosDIS, Hubble Space Telescope, CDOS [11]). NASA has realized that, with recent advances in computerized information technology, there are opportunities to enhance the useability of collected data. There is an understanding that data and information management systems required to support the Earth and space research community will need to be complete research information management systems [10]. One fundamental requirement of these new systems is to allow flexible and transparent access to geographically dispersed, heterogeneous databases. Access methods will need to use advanced information management techniques including appropriate search and AI methods. The goal of such systems is to elevate the scientist from having to concentrate on the details of accessing and preparing data for analysis to concentrating on his research [10]. In response to these problems, NASA has begun the Intelligent Data Management Project to begin research into and implementation of advanced information management systems [14]. In addition to this NASA effort, many other researchers have begun to address issues surrounding the IIMS.

3.3. Intelligent Information Management Systems

The goal of future Earth and space information management systems is to intelligently manage data in four areas: (1) data collection, (2) data processing, (3) database management, and (4) accessing and archiving [9]. In light of the total information system perspective, research reported in this paper has concentrated on the development of intelligent user-system interfaces and intelligent information integration and access. Following sections will focus on these areas. However, before discussing features of the IIMS, it is important to understand some of the more salient limitations of current information or data systems that provide for computer access. Typically, such systems rely on traditional database technology that has several disadvantages when providing for a complete research system. Some of the more important limitations are discussed in the following sections.

3.3.1. Query Limitations

The method generally used to extract data from current systems is querying the database through the use of a query language. To query the database, the user must *instruct* the system concerning the scope of data in which the user has an interest. After forming a sentence recognized by the particular query language in use (which, in general, is not natural language-like), the database system will return the requested data; the user may or may not be able to interactively view the data. In either case, the user must then determine, after some amount of processing, if the data satisfies his needs. Such a process may be repeated several times before data satisfying the user's requirements are obtained. The process of forming a query statement and then analyzing the data returned from the execution of this statement has several limitations.

Learning the Query Language. Before obtaining data from the database, the user must learn how to interact with the system; such learning is accomplished through the use of a query language (i.e., normally mathematically rigorous grammars that require the user, when requesting data, to express boundaries in a rigid syntax). These languages are typically difficult to learn, and their effective use often requires a high level of expertise. The user can easily become confused about how to express requirements for data sought, and frequently, needed data cannot be located, or data that is not needed is received because of difficulties inherent in the forced, unnatural expression of the request.

Knowledge of Data Structure and Relationships. After the user has learned the query language to a degree which permits forming queries, the next step in the process is comprehending the structure of the database. This is made necessary because the query language requires that the user inform it where the data resides within the database structure. In a relational system, the user must know the names of relations and attributes by which data is stored. A second limitation is that data in these systems are usually organized for speed of access and without consideration either for relationships among the data or concepts inherent in the domain of the research. Therefore, to access data relating to a basic concept within any research domain might require a single query spanning several relations and performing many calculations. When using all but small databases, such a requirement places a large burden on the user, and its execution is usually beyond the capabilities of researchers dealing with large applications. For example, the Crustal Dynamics Data Information System database consists of 144 relations and 679 attributes [14]. To use a normal query language, the user would have to know all relation and attribute names, as well as relationships between attributes, and be able to translate domain concepts into the query language. Obviously, to do so for the Crustal Dynamics Data Information System database would be extremely difficult.

Query Reformulation. In some cases, queries will not extract the expected data, requiring the user to resubmit a modified query which, hopefully, will return better results. Typically, no aid is given to the user in comprehending why the original query did not return the required data, and thus, it

may not be clear how to appropriately modify the query.

3.3.2. Interface Limitations

In addition to limitations discussed above, current database systems often fall short of providing needed capabilities in the area of user-system interface. Systems today are often text-oriented with little facility for graphic representation or direct manipulation of screen objects. User-system interface requirements deemed necessary for an IIMS are detailed in section 3.4.2.

Beyond limitations already discussed, there are other querying process and user-system interface technology limitations of current systems. However, the discussion provided above has shown the general unsuitability of a traditional approach for the large, complex databases like those that are and will be encountered in Earth and space research. Rather, new capabilities and new approaches are required.

3.4. Functional IIMS Requirements

Several other information management concepts and technologies have emerged over the past several decades that can be brought to bear on current system limitations. However, before we can evaluate the appropriateness of any of these approaches, a formal definition of IIMS capabilities must be presented. Our effort has concentrated on how to provide the user with the capability to locate and manipulate desired information while at the same time allowing an efficient and flexible mechanism for providing automatic updates to the information data/knowledge base. The IIMS functional description presented in following sections reflects this conviction. Requirements for an IIMS that our work has addressed can be divided into two categories: (1) foundations of the system; that is, those parts of the system not directly accessed nor seen by the user, and (2) the intelligent user-system interface, through which all interactions between the system and the user are managed.

3.4.1. IIMS Foundations

The foundations of any system which accesses data are its underlying information structures and its methods for manipulating those structures; this is particularly true for the types of large systems we are addressing. In these systems, data will likely be stored in heterogeneous computing environments. The access system will have to be able to handle multi-users and allow timely updating of the various databases and knowledge structures without interrupting users of the system. Because of the significantly large amounts of data entering the IIMS on a daily basis, updates to data and knowledge bases will need to be automatically performed. Any attempt to handle such updates using manual techniques would soon overwhelm available personnel. Some projected systems will require that data be incorporated into the system and made available to users within a few hours. Meeting these requirements depends on the design and implementation of information structures and intelligent techniques allowing for automatic update.

Access to heterogeneous databases and information structures must be transparent to the user. To provide transparency, the query subsystem must be able to decompose a query the user has formed into a set of queries that access appropriate databases in their associated query languages. This subsystem must be able to manage multiple sources of similar data types and, depending on the cost of accessing each alternative and the loads on and limits of respective systems, access the most appropriate database. Additionally, queries might involve data processing by special algorithms. The query subsystem must understand where copies of such algorithms reside and processing limitations of the various host systems. If part of the data is not available, or the cost of query execution is above a certain threshold, the user should be notified. If a query can be executed, results returned must then be combined and processed to provide the answer to the original query, and then converted into the correct presentation form. Future information systems are currently planned as long-lived systems, and it is generally accepted that they should be able to adapt to short- and long-term variations in operating conditions and operate with low maintenance costs. To satisfy these conditions, data and knowledge structures must be able to be modified to interactively incorporate new types of data without interruption of normal system operation. Any inflexibility in this respect could render the system unusable when operating conditions change.

3.4.2. Intelligent User-system Interface

The role of the intelligent user-system interface is critical to the success of the entire IIMS. Even if the foundations are successfully laid, it does little good if the user is not able to easily and efficiently access information. Within the normal pursuit of research goals, the user-system interface should stimulate the user to explore and use the whole system with all its capabilities, and should always be an aid, not a hindrance, to the user. While these goals have proved difficult to satisfy in a general sense, much has been learned from research in their pursuit. Some features discussed in following sections have been shown to be useful in achieving these goals, while others are just now beginning to be explored.

User Profiles. One technique that can contribute to providing a supportive interface is using knowledge about the user to make system access more efficient. Most generally, each user has different areas of interest and levels of expertise (both within the user's domains of interest and knowledge of the IIMS) that can change over time. To provide the best environment for different users, the system must be able to adapt to such variable conditions, and also understand information about individual users or groups of users. We have found it useful to group such information into two categories: context sensitive expertise and goal

comprehension.

To provide ease of use for experienced as well as novice users, the system must understand the level of domain and system expertise at which the user is operating. A system may be easy for a novice to use but may be very difficult, for various reasons, to use for an expert. In particular, as a user becomes familiar with certain parts of the system, support tools necessary for a novice user are no longer required. For example, a menu-based system may be adequate for a novice since it provides a type hierarchy for locating available operations. However, menus can be time consuming for an expert unless shortcuts, such as commands bound to keys or macros, are available. Despite allowing for some limited user growth in system operations, menus alone are not sufficient for the type of system proposed here since they do not recognize the expertise of the user. To be truly effective and useful, the system must understand the expertise of the user within the context in which he/she is currently operating; understanding which is required since users' expertise varies for different parts of the system. To address user expertise by measures such as time-of-use is simply too course grained. The growing subsystem operating expertise of the user must be reflected by modification of offered support features.

Future information management systems will contain many different types of data, ranging from accounting and collected scientific data to information about instruments and users. As a consequence, the system must be able to accommodate operating variations resulting from users with different interests, goals and expectations about using the system. Two particular concepts are useful in understanding the need to manage such variation: intelligent views and group-type hierarchies.

Intelligent Views. For users to be most comfortable with data access and presentation, the IIMS must be able to communicate with the user using familiar terminology, concepts, and data organization. That is, the system must relate to the user in a manner consistent with the user's domains of interest. Intelligent views are one technique used to incorporate knowledge about the user's domain and concepts of data organization. For example, a user who is interested in cost figures for certain organizations, or performance of the system, is not necessarily interested in all data the system contains; presenting all data would only serve to confuse and inhibit the user. Hence, only data or information within the user's current domain of interest should be displayed. In addition, information should be presented to the user in a manner consistent with the type of data being displayed (e.g., accounting information being displayed in the form of graphs or charts). Users may select standard views defined for various application areas, and the system should allow existing views to be modified or new views to be created.

Group-type Hierarchies. Since interests and goals tend to be related within groups of users, the IIMS should be able to handle a user group-type hierarchy and associated inheritance capabilities. This would allow for users to inherit certain characteristics of a particular group (e.g., science users, system operators) while allowing for personalized modifications.

Help Facilities. Another feature required to provide a complete IIMS is an extensive *help* facility. *Help* should be available no matter what the user is attempting to accomplish, and all *help* should be context sensitive. For example, a general explanation of the current system should be provided when no specific subject is selected, and when a specific subject is selected, detailed information should be provided. *Help* facilities must be responsive to levels of user expertise in the selected subject and should allow for exploration of related information. A general overview of information concerning system operation and data available should be provided to novice users (tutorial information), while expert users are immediately allowed to select more detailed information. The novice user can progress to more advanced operations by navigating through the information base associated with any particular subject area. There should be no difference in presentation style and methods used to provide *help* about system operations or *help* about data. For example, if a user requests more information about a particular instrument, the user should be able to navigate to a drawing of the instrument, descriptions of its specifications, other related instruments, and possibly information about the manufacturer. Information required to answer users' requests for *help* is structured no differently than any other information stored in the information base. Finally, *help* facilities include an intelligent tutorial system (see section 4.3.3) which can fully explain the IIMS and its capabilities. This tutorial system is linked to the rest of the *help* system and uses much of the same information as other *help* facilities, allowing the user to obtain a tutorial on a current operation without specifically entering the tutorial subsystem.

Intelligent Information Manipulation. The IIMS must provide an interface to the system's query capabilities, and should allow the user to transparently form queries without having to learn or use a mathematical query language. This type of query formulation system insulates the user from having to know (1) the structure of the data and (2) what information is contained in the database. As the user forms the query, the system should indicate what data is available at each stage of the query formulation process. The user should be allowed to form queries using conceptual information processing techniques provided by intelligent views. Using this methodology, the user should be able to form a query by inserting range requirements, spatial, and temporal indications. Moreover, the IIMS should be able to store and provide for query modification and reuse, as well as allow for browsing the information base. Ideally, browsing and query formulation capabilities should be combined. Finally, the system should be capable of displaying different

types of information in graphic formats.

3.4.3. Flexibility and Extensibility

As stated earlier, data and knowledge structures must be able to be modified to handle new data types without interruption of normal system operations. Furthermore, the user-system interface must be flexible and extensible. Different users have different preferences for the organization of the interface and use of available system support tools, and need to be provided with a measure of control over interface presentation. Finally, the capability to access algorithms and other external programs (e.g., Geographic Information Systems) must be allowed without the need for reprogramming. In general, flexibility and extensibility of the system must be carefully considered in all design and implementation decisions.

3.5. Other Information Management Techniques

Research in information management has resulted in many useful systems and techniques. Each of these systems and techniques has focused on certain issues or types of data but none have addressed the entire range of IIMS functional requirements. Database research has provided several techniques and systems, including (1) the ability to efficiently provide multi-user access to distributed databases, (2) indexing techniques for various types of data, including spatial and temporal, (3) transaction processing and recovery techniques to ensure data integrity, and (4) dynamic data management. Hypermedia systems provide information and techniques on (1) navigation through a large information base, (2) integration of various types of media, and (3) representation of relationships among data. Finally, AI systems have provided valuable techniques for (1) representation of relationships among data as well as other knowledge representation capabilities, (2) data driven processes, (3) planning and scheduling systems and (4) intelligent user-system interface capabilities.

Taken alone, none of these approaches can satisfy all requirements for the next generation IIMS. In general, proponents realize the limitations of their approach and are beginning to work on removing these restrictions. Database researchers, for example, are examining ways to incorporate data relationship information while continuing to research new indexing techniques for different types of data. The field of hypermedia is beginning to examine incorporation of search techniques to allow for more efficient navigation within the information base. Expert database systems are studying the use of indexing techniques to improve system performance. All these approaches are addressing the problem of accessing information in an efficient manner, and although they began by looking at different types of data, research directions are resulting in movement toward a common viewpoint. We believe that the next generation IIMS should combine experience gained in each of these fields of study; not an easy task since the approaches are not always compatible and some projected capabilities of the IIMS are still beyond current technology. However, the use of appropriate techniques from each of these approaches, combined with lessons learned from user-system interface research, should provide a good starting point for implementing such a system. As discussed above, we have taken an eclectic view of the IIMS and the following section details our current approach to developing the IIMS prototype, and our present thinking concerning several issues surrounding future IIMS development in relation to the chosen application of large scientific databases.

4. APPROACH

4.1. IIMS Operations Concept Development

Advanced information system operations environments will be required to perform and provide support for a multitude of functional capabilities, ranging from data capture to teleoperations. Future information system design will be largely influenced by the demand for extensive and sophisticated data acquisition, handling, processing, management, and access facilities. It is critical that data not only be obtained in a timely and cost-effective manner, but that acquired data be made readily available to potential users. While issues associated with efficient data acquisition are significant, it is the latter requirement that necessitates looking beyond traditional and currently available technologies. Issues of user transparency, in terms of command and control, planning and scheduling, processing, and information management and access demand application of new and innovative computational programming paradigms.

Several potential factors influence advanced information system operations. For some functions (e.g., command and control, planning and scheduling), the information system may provide a direct link between the user and the flight system. It is important to realize that cost-effective future information system operation will require that many current, man-intensive operations functions will either become part of the flight system functional repertoire or will be handled autonomously on the ground. Again, command and control, planning and scheduling, system management, and fault detection/isolation are excellent examples of potential automation candidates. Future information system features of telescience, user transparency, and remote distributed access will require levels of system operation sophistication not currently available. The purpose of our work, then, has been to closely examine several projected, advanced information system capabilities in light of the potential application of advanced programming technology.

As suggested earlier, it seems apparent that two major issues confront the design and implementation of the IIMS: first, how to design an effective, transparent user-interface which not only allows for multiple remote user access using different hardware/software but also provides intelligent support features for helping users with different levels of expertise

explore, identify, and obtain appropriate system data, and second, how to design an efficient and flexible mechanism for allowing automated incorporation of (1) new data types, (2) information about the new data, and (3) appropriate links with other metainformation. The prototype IIMS will provide a proof-of-concept demonstration of the applicability of AI programming techniques and tools to the domain of distributed data access and management. The prototype IIMS will be able to (1) accept multiple user data requests from different types of remote user-interfaces, (2) intelligently guide the user to an understanding of the types of information and data available, and (3) accept and integrate information concerning data updates from multiple, heterogeneous databases.

4.2. Application of AI and Rapid Prototyping to the IIMS

Before discussing our approach to determine the potential applicability of AI technology or advanced programming techniques to advanced information systems, it is important to understand some basic assumptions we are making concerning AI:

Assumption 1. The term 'AI' is not particularly useful for describing the many potential advanced programming technologies which may be applicable to advanced information systems. It would probably be more appropriate to determine the potential applicability of AI in terms of more specific sub-disciplines such as expert systems, intelligent resource allocation, intelligent user-system interfaces, intelligent networking, intelligent training, intelligent information management/access, simulation, automated programming, natural language, and machine learning.

Assumption 2. It is important to distinguish between (1) application of AI technologies to advanced information systems and (2) use of advanced programming techniques (e.g., object-oriented programming, production systems) associated with AI to facilitate development of more traditional types of applications. For example, work of ours has shown an almost eight-fold decrease in code development time and an almost four-fold cost savings when using object-oriented programming and production systems to develop simulations of Spacelab payload experiments for purposes of payload crew training [3].

Assumption 3. Although initial decisions concerning potential advanced information system candidates for AI technology application can be made using specific, paper/pencil criteria, rapid prototyping will likely provide a much clearer answer concerning the potential adequacy of certain technologies for information system applications. This is particularly true when attempting to apply advanced programming techniques and environments to more traditional types of programming applications.

Assumption 4. Although we believe appropriate application of AI technologies/advanced programming techniques would provide significant benefits to certain areas of advanced information system operations, we also believe that such technologies should be applied *only* when appropriate. Often, a hybrid of traditional and AI programming approaches is useful for optimal system performance as well as system modification and enhancement.

Our basic approach to determining the potential applicability of specific AI technologies and advanced programming techniques to advanced ground system functionality is highly iterative, with domain experts involved at each step of the process. Using surveys of existing systems, initial criteria, and expert advice, potential application candidates are culled from the total domain and after further examination, a select number of applications are chosen for proof-of-concept demonstrations. Rapid prototyping these applications using AI technologies and advanced programming techniques is, again, an iterative process, serving both to identify candidates for system implementation as well as to help drive out more detailed system requirements. An important feature of this process is that initial candidates can be quickly chosen, and application of rapid prototyping techniques provide the mechanism for rapidly narrowing the potential field of candidates for actual system implementation. Of course, the type of prototype development environment used is crucial to the successful application of these techniques (section 4.5 describes the hardware and software configuration we are currently using for our effort). Given results from the prototyping effort and input from domain experts, a final set of applications is chosen for actual system development.

4.3. IIMS Prototype Implementation

To date, our work has suggested several potential IIMS functional requirements that provide good AI technology application candidates. Figure 1 illustrates our current view of basic IIMS functionality and indicates those areas we believe will most likely benefit from the appropriate application of AI technologies. Although our prototyping work has focused on a limited set of these, we believe that all are appropriate areas where advanced techniques could be applied. Discussion of our present thinking concerning implementation of each function is given below.

4.3.1. User-system Interaction

Rather than a set of separate modules which are activated when required, we view all interaction between the user and the IIMS to be handled by a single, integrated system. All user-system operations are physically and conceptually linked to the system as a whole, and as a result, *help* facilities are handled in the same manner as browsing the data or knowledge bases. Such a technique maintains a consistent user-system interface and simplicity of design, while making it easier for the user to learn and use the system. When all data is treated the same, accessed in the same manner, and traversal from one set of data to another is transparent, the

user does not have to learn different methodologies to interact efficiently. In addition, context switching is not dramatically experienced, thus reducing confusion normally associated with such transitions.

To provide rapid access to an environment whose structure can be transparently and dynamically altered, we employ a conceptual structure of *metadata*, or information about the data, as the main information structure. All information including non-scientific data (e.g., *help* information, tutorial

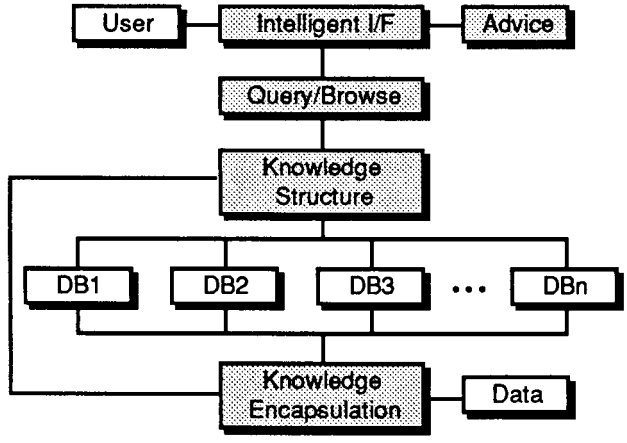

Figure 1. IIMS Functional Architecture

information) and information about the scientific data contained in the system is stored using the same conceptual structure. Actual science data, stored in distributed databases, are accessed only when a query is being resolved or specific information about the data is required. *Metadata* structures are automatically generated and maintained by a set of expert systems which incorporate conceptual information about different domains and relationships among data. All information base (real data and other knowledge) update messages are processed through these expert systems which then make appropriate modifications to the knowledge structure. We choose to use this approach so that navigation of the information structure will not be slowed by the execution of a large expert system; we can still use the powerful techniques expert system technology provides for automated knowledge extraction and information updates.

For users to be most comfortable with the task of locating information, data presentation must occur in a familiar manner. Appropriate *metadata* structures should accurately reflect domain concepts and use domain terminology. As viewed by the user, such structures cause domain specific variation in the knowledge structure, although no change occurs in basic relationships among data which form the global structure. Providing this capability, relationship information is provided by physical, typed links with concept dependant, relevance weightings, allowing for representation of the notion that certain links have a high degree of relevance for certain domains and have no relevance for others.

As the user traverses the knowledge structure, links available for selection by the user are chosen based on (1) the relevance weighting factor within the context of the current domain and (2) the path the user took to arrive at the current node, and (3) a user defined threshold value. Since weighting factors allow for a range of values between 0 and 1, a wide range of concepts can be represented, allowing for the inclusion of fuzzy concepts and searches. We will use relevance weights to provide for inclusion of relevance factors to represent different levels of user expertise.

We provide mechanisms, which we call intelligent views, that allow the user to define and modify knowledge structures (In the future, they will also allow for the specification of plans and other intelligent support features). The user can then select which view is considered most appropriate for the current task and activate it. This modified view is then layered on top of the global knowledge structure (this does not alter the global information base; only the user's view of it), changing the user's view of the global knowledge structure and providing the capability to explore the data in a familiar manner. Intelligent views can be shared by users, allowing the facility for domain experts to define a series of views to be used by others. The method for creating these views is the same as that used in navigating the knowledge structure, with a few extensions to allow for (1) creating new or combining already existing nodes, (2) including plans or other knowledge based programs, and (3) altering weighting factors.

Query formation is achieved by indicating items of interest through the user's navigation within the knowledge structure. As stated above, choices offered to the user at each location are determined by calculated relevance factors. At certain locations within the knowledge structure, users can indicate Boolean and range specifications for certain attributes, with potential extensions to allow for fuzzy characterizations. The manner in which range specification is indicated depends on the type of attribute. For example, a map of the world is displayed to allow the user to graphically specify the spatial area of interest. The user is also allowed to select predefined regions (system or user defined) or, if desired, enter map coordinates. Facilities for scrolling and zooming provide the user with desired levels of detail. With his methodology, there are no perceived differences to the user between *help* and scientific information. Both are information for which links have been defined. Therefore, context sensitive *help* is provided in most locations by links to appropriate information.

Using navigation techniques to allow formation of queries eliminates many of the traditional problems associated with standard query formulation: first, the user always knows what data is contained in the database and what next selections are possible. Second, since this process is constrained to knowledge the system contains, only valid queries can be formed. Finally, the user does not need to learn a query system but can "converse" with the system in the

terminology of the chosen domain. As a result, this method provides facilities to query the database and browse through the database, locating information of interest or exploring related topics. While providing several advantages, there are also problems with this type of query formulation. Four we are currently concerned with are (1) the problem of getting "lost in space", a problem encountered by hypermedia systems when navigating a large body of information, (2) elimination of extraneous operations encountered by users familiar with the data, (3) providing non-navigation movement techniques, and (4) smooth integration of Boolean and range qualification specifications.

Lost in Space. When users are allowed to navigate a large body of information which provides a rich environment for exploring related topics, disorientation, due to inadequate cues concerning current location, is typical. Standard graphical techniques that display the information base as a group of nodes and links can become unusable because of the density and complexity of displayed links [4]. We are using several techniques to address this problem. Intelligent views help reduce the information space and the amount of information directly accessible at any one time, since not all links proceeding from any particular node are relevant to the current domain context. This technique aids in reducing complexity, but, in many cases, will not reduce it enough to allow for total reliance on graphical display techniques. One method we are exploring is to provide graphical structure overviews to show detail based on both distance and relevance. Consequently, the farther away the user is from a node, the higher the relevance threshold is for links, thus reducing the number of links displayed. Furthermore, there is an inherent, hierarchical organization in much of the information space that can be utilized, aiding the selection of nodes to be eliminated from the display. As a result, the number of a node's "children" (this term is used loosely) not displayed depends on their distances from the current node, appropriately reducing the amount of information displayed while providing enough location aids to be of use.

Elimination of Extraneous Operations. Users tend to frequently state the same or closely related queries. This inclination could become cumbersome if the user was forced to traverse the information base in the same manner each time; the user would be forced to frequently enter repetitive operations and would probably become disillusioned with the system. We are addressing this issue using several alternative methods. The user will be allowed to save partial or complete queries that can be displayed in graphical form, modified, and then resubmitted. These saved queries can be treated as an intelligent view by highlighting query choices on the backdrop of the active view at the time of query creation. Such a procedure allows the user to modify specific query values or include or delete other items.

Alternative Navigation Techniques. While navigating through the information base at the same time as forming a query or browsing, the user may wish to move to another location that cannot be directly accessed from the current location. The absence of this capability would force the user to find some path through the information base, a time consuming and often difficult process. Initially, we will allow two methods to provide this facility: (1) via the mouse, allow selection of a node from the graphical knowledge structure display, and (2) allow the user to enter a keyword or phrase as a search criteria, resulting in a display of all nodes satisfying this criteria. The user will then be able to choose any appropriate node.

User Profiles. We have already discussed some potential types of user or group information that can be defined and manipulated (e.g., intelligent views and saved queries). This information along with other information about the user (e.g., group type, domains of interest, expertise when dealing with different parts of the system, interface preferences, accounting information, access privileges) will be kept in a user profile. Defining the user for the IIMS so that the system may be appropriately configured will aid in providing easy and useful user-system interaction. In general, users will be considered as objects and user hierarchies will be established, allowing for inheritance of abstract user information while providing for individual user preferences.

Results Display. The final part of intelligent interaction with the user is the display and manipulation of results. At this point, it is unclear what types of displays and interactions would be most useful. It is understood that graphical displays, including image displays, will be required, but the extent of potential results display functionality has yet to be determined. It is fairly obvious that much depends on the type of data to be displayed (we have already alluded to the notion of display representation being closely tied to the type of data being displayed). We plan on interviewing potential users of these types of systems to determine what types of display are required.

4.3.2. Knowledge/data Encapsulation

All data stored in IIMS databases are input from various sources, and when new data becomes available, the IIMS catalogs and directories must include information about the new entry. This process is accomplished through the mechanism of an update message which informs the IIMS that new data exists, where it resides, and a number of appropriate attributes which describe the data. Additionally, information concerning associations and relationships between the new data and existing data must be generated. In general, the knowledge structure describing data that exists must be modified to accept the new information. Clearly, given expected high data rates and volumes, it would, at best, be inefficient to process such metainformation in a manual fashion. Hence, the requirement for an automated *metadata* update system is essential.

Upon receipt of the message indicating new data has been input, new *metadata* is processed and inserted into the IIMS

*metadata*base structure. As suggested above, such processing involves understanding the new data so that appropriate relationships between new *metadata* and other *metadata* already contained in the *metadata*base can be established. The speed at which this process can be accomplished depends on (1) the type of data received and (2) its potential relationships to other *metadata*. Establishing these links can be technically challenging, and the success of any initial capability and the subsequent learning of future relationships from patterns of use during user-interaction sessions is key to efficient *metadata* access. The most promising approaches to handling metainformation inference seem to be closely allied to object-oriented database technology [7]. We expect that one or more expert systems will be required to provide the inferencing capability required for such a system. Initial estimates of potential data input frequency and associated update messages will need to be made since it could be possible that infusion of new data to the IIMS will outpace its ability to accomplish necessary processing of the *metadata* in real-time. The IIMS will therefore require the ability to buffer update messages until such time that processing can take place. An alternative solution is to immediately update the IIMS's databases and perform required *metadata* processing on a delayed basis.

Moreover, *metadata* extraction algorithms will have to be provided to the IIMS for each of the types of data the IIMS must know about. Since most of these algorithms will be application specific and subject to modification, a method must be established to handle this dynamic environment. New or modified algorithms will need to be incorporated in an efficient manner. A system will be required which allows specific format specification of the *metadata* to be received by the IIMS, data attributes which are important, attributes to pass on, any processing which must be applied to the attributes to produce additional attribute information (e.g., algorithms will be required to extract relationships among attributes). In some cases, *metadata* updates will need to be manually submitted when *metadata* cannot be automatically generated (e.g., the user may wish to suggest some alternative associations or relationships that are not obvious to the system). To make this process as efficient and consistent as possible, an intelligent forms entry system will likely be required.

The representation sophistication of data relationships directly influences the type of implementation necessary for IIMS data- and *metadata*bases. In the simple case, where only a few links are provided, a commercial database, along with a simple conceptual net or other knowledge representation technique, would probably suffice. At the other extreme, all data could be stored in a knowledge based system. The problem with the second alternative is that such a system may not be able to efficiently handle the volume of data projected for the future IIMS. The most probable implementation will be a commercial database to efficiently handle volume and query processing in addition to a knowledge representation scheme/expert system which helps determine relationships among the data. These two systems would work closely together to provide desired capabilities. It is not clear at this point just how much responsibility would be assigned to each system.

4.3.3. Advising/tutorial Capability[1]

Since the primary consideration used to determine the usefulness of any information management system is its ability to make information accessible to the widest variety of personnel, it is extremely important that the system be easy to use and provides the necessary user support. Several potential user support requirements have been postulated [2], including:

1. Facilitating understanding by specifying the contents and meanings of a collection of data as well as the relation between objects within the data;

2. Supporting approximate reasoning to infer conclusions that are not explicitly stated by the user;

3. Handling information demands for which the database is used routinely (macros);

4. Communicating with the user in plain English text;

5. Providing a physical and logical link between information contained in the meta-knowledge database and the actual data.

As indicated in section 3.4.2, we might add to these the notion of intelligent *help* or tutorial capability, allowing naive and expert users ease of system use without long learning times. Indeed, the intelligent *help* facility could either be on- or off-line, depending on the mode of operation in effect at the time. User profile capability would be very useful here, being used to expedite transactions and containing at a minimum the following user information: skill level in using the system, user areas of interest, conceptual views the user employs, type of terminal and software the user is operating, location of the user, communication lines which are available to send data, and accounting information. As with all *metadata*, the IIMS should be able to continually update this information as the user learns and interacts with the system, allowing interaction methods to be adapted as user needs or skill level changes. Our IIMS prototype implementation ties the user profile capability to the intelligent *help*/tutorial facility to allow more effective system use for each individual user. As outlined in section 4.4, using objects as a user-model representation technique provides exceptional flexibility in facilitating overall IIMS operations.

---

[1] A detailed discussion of Intelligent Tutor Systems is beyond the scope of this paper. Excellent overviews of the field can be found in [13,17].

In terms of actual advising, it is important to explicitly deal with the user's conceptual model of the IIMS [5,12]. Obviously, each user may have a different conceptual system model, resulting in confusion if the underlying system is not equipped to handle the varied expectations. Such conceptual model information could be associated with the user profile and used in several ways (e.g., design of user views, complexity and types of data/knowledge display techniques, query capability). Direct manipulation interfaces often provide the most effective means for learning a new system and helping expert users with ordinary day-to-day tasks (e.g., allowing the user to directly construct command macros used to speed system operations). The current IIMS prototype implementation relies heavily upon direct manipulation of system features to provide the user with necessary information and *help* concerning operations.

4.4. Object-Oriented Programming and the IIMS

The notion of representing dynamic systems as collections of interacting components with associated behaviors is intuitive, and provides a basis for designing more flexible and easily modified software systems [3,8,16]. Message passing among objects emulates potential system interactions since single messages can initiate any number of complex behaviors or other messages. Clearly, use of objects need not be limited to simple system component representation. They may also be used to represent other, more abstract processes and procedures (e.g., control processes, commands, *metadata* templates). Object-oriented programming paradigms are rapidly becoming a primary software development methodology used in a wide variety of applications. In particular, recent work in database and database management system (DBMS) technology has begun to seriously consider object-oriented data structures as a viable alternative to more traditional data representation techniques. Three primary advantages of object-oriented data models, when compared with more traditional models, have been postulated [6]:

1. Databases can be viewed as a group of objects rather than a set of relational tables;

2. Object-oriented data models support at least two types of data relationships: interconnections among data and generalization or subtyping of data types;

3. Integrity constraints are more easily implemented using object-oriented paradigms.

Additionally, there is an increasing trend to develop data models consisting of sophisticated, hierarchical data constructs that often share attributes or functional characteristics, resulting in highly portable, semantically-oriented database systems with several possible categories of relationships among objects (e.g., generalization [is-a], aggregation [a-part-of], and association as well as relationship qualification [1,18]). As suggested earlier, using objects as both a data and knowledge representation technique provides a data and information management system structure allowing for semantic interpretation of the data residing within the database. In addition to using objects as a representation technique for databases, several other IIMS functional requirements lend themselves to object application.

Displaying information is crucial to the ability of the user to understand not only what is in the database but also the relationships among the various types of data. Recent developments in object-oriented display systems (e.g., windows, graphic objects) are well suited to projected display requirements of advanced information management systems. As described in section 5.0, we have implemented an interactive graphics object system to facilitate both the display and representation of complex data and *metadata* structures. The system is used extensively in the prototype's query implementation and will be used to develop both online and tutorial capabilities as well as in the area of data presentation.

Implementing a system that can effectively respond to any particular user's needs is a considerable challenge. We have already stated the requirement for the IIMS to provide individual users with personalized environments in which to work. Object-oriented programming provides an appropriate structure for supporting such individualization. Since we anticipate that future IIMS users will fall into natural user hierarchies, it is useful to represent those users as an object hierarchy. Of course, almost unlimited user information may be associated with appropriate user objects, allowing for efficient and more effective user-system interaction. In particular, user views can easily be designed to reflect individual needs without fear of adversely affecting any other users.

4.5. The Mission Operations Laboratory

The current Mission Operations Laboratory (MOL) configuration is shown in Figure 2. As indicated, current programming is entirely microprocessor-based. The choice of a microprocessor-based computational and programming environment for the MOL was predicated on two major considerations: (1) the growing trend toward microprocessor workstation technology and (2) the recent availability of advanced, highly sophisticated dual and multi-processor micro systems. Our intent in designing the current MOL configuration is to provide an environment similar to those we expect to find in the next generation of information management systems, many components of which will be based on heterogeneous computing environments. The advent of true telescience will bring with it the capability to allow remote users, using many different types of hardware and software, transparent operations and data access. To provide the extensive user support required for efficient operations, high-level languages (e.g., Ada, LISP, Prolog) will need to be supported. Advanced, non-traditional programming

paradigms will likely become commonplace since they provide the flexibility required to handle significant software modifications without significant impact or perturbation to system operability. In addition, application of AI technologies to several system functional capabilities necessitates powerful software development environments.

In the near future, the capability of the MOL to test various advanced concepts in the domain of information management will be significantly enhanced with the addition of hardware using both parallel processing and advanced optical disk technologies. First, the ability to apply different parallel architectures to features of an information management system will lead to a much better understanding of how such topologies might influence areas of user-system interaction as well as data handling, storage, and access. As indicated in Figure 2, our choice for a parallel development

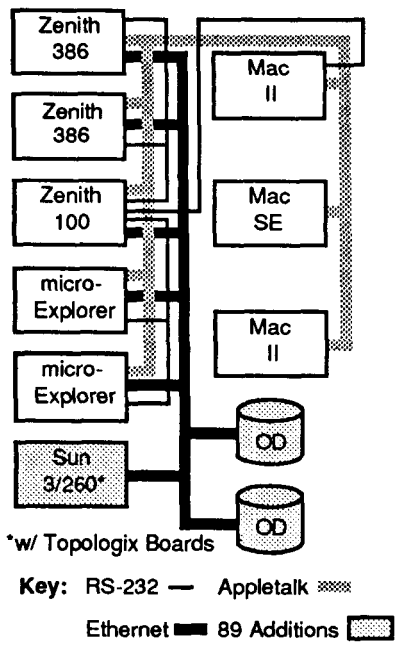

Figure 2. Mission Operations Laboratory

environment is a Sun 3/260 including multi-processor capability using transputer technology. The resulting message passing architecture is ideal for proof-of-concept design and test of several parallel topologies applied to any number of information management system functional domains.[2] Second, optical disk technology provides the capability required to test advanced knowledge and data storage concepts. Projected information system high data input rates and volume storage require that the impact of optical disk technology on both functional and performance requirements be carefully examined.

4.5.1. AI Microprocessing Environments[3]

Before 1988, microprocessor hardware capable of meeting current and projected requirements for AI computing did not exist. In 1988, Texas Instruments made a microprocessor system available that hinges on the recent development of technology allowing LISP processing on a single chip. The microExplorer™ provides most of the programming environment found in the parent Explorer machine and, similar to all dedicated LISP machines, is object-based. The microExplorer is hosted on a Macintosh II, with the microExplorer providing a dedicated LISP processing environment and the Macintosh II providing the I/O and user-interface. Functionally, the LISP processing environment is handled by the Macintosh as it would any other application. Indeed, other applications (e.g., Hypercard) can run simultaneously with the LISP processor. MicroNet™ and NuBus™ serve as the interface between the microExplorer and Macintosh II environments. Using either the microExplorer or the Macintosh II, capabilities exist for creating application programs that service the other processor. The Network File System™ (NFS) is an RPC server composed of several procedures and provides transparent file I/O capabilities for operating systems if requested. The eXternal Data Representation™ (XDR) protocol provides common data representation across networks.

4.5.2. Traditional Microprocessing Environments

As described earlier, future information management systems will be required to operate in environments where heterogeneous computing systems are the rule rather than the exception. The MOL provides for heterogeneous microprocessing capability by including, in addition to the microExplorers, traditional microprocessor facilities. Two Zenith 386s provide more standard OS/2 operating systems, and two standalone Macintosh IIs provide two different operating systems: one uses the standard Macintosh operating system and the other A/UX. Of course, both Macintosh IIs, as well as the microExplorers, provided extensive graphics capabilities. All MOL hardware is networked internally with the resulting LAN networked to the Information Network Laboratory.[4]

4.6. IIMS Prototype Development

Implementation of the IIMS prototype will proceed in a

[2]The Topologix™ configuration may be dynamically reconfigured at runtime to emulate any of several parallel topologies. Additionally, hybrid architectures may be designed with one topology applied to one aspect of the application and a second topology being applied to another part of the application. Currently, parallel versions of FORTRAN, LISP, and C will run on the system.

[3]The Remote Procedural Call (RPC) and eXternal Data Representation (XDR) servers and the Network File System (NFS) are products of Sun Microsystems. NuBus and MicroNet are Products of Texas Instruments.

[4]The Martin Marietta I&CS Information Network Laboratory provides a flexible networking environment for proof-of-concept research, architecture design, and prototype development in the areas of tele- and data communications.

highly modularized fashion, with early iterations of functional modules representing simplified versions of later enhancements. Overall, we are planning for four phases of prototype development.

4.6.1. Phase 1 Development

Figure 3 shows our initial concept of projected information management functional allocation to software modules. During Phase 1, rudimentary versions of all functions will initially reside in one of the microExplorers. Briefly, the *User* function initially represents an abstract simulation of the 'typical' IIMS user; we anticipate several categories of potential IIMS users. However, while the prototype will

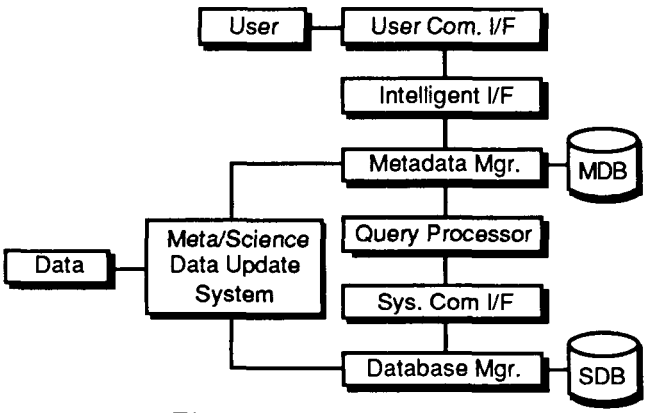

Figure 3. IIMS Software Allocation

examine functional support features for many of those categories, we will initially focus on the science user and support functions he will require. It should be apparent that many of the support features developed for one type of user may be generalized to other user types. Another important aspect of the *User* function is to allow remote access. Phase 1 *User* capability will not allow for remote access but will focus on the types of required user-support features. Two *Communication Interface* modules, one to handle communication between the *User* and the IIMS *Interface Manager* and the other which handles communication between the *Query Processor* and the *Database Manager*. The *Interface Manager* is the module which handles intelligent support features for the user. For example, information on user profiles and intelligent user views would be kept in this module, allowing for individual system configurations when the user logs on. An important feature of this component is allowing the user to modify his user view when appropriate, thereby 'modifying' the underlying *metadata* structure found in the *metadata*base (see section 4.3.1 for more detail). Additionally, all on-line *help* facilities would be included here, and the *User* would be able to create procedural command macros allowing for more efficient information navigation and access. The *Metadatabase Manager* interacts closely with both the *Interface Manager* and the *Query Processor* helping to direct the user to appropriate information without wasting an inordinate amount of the user's time. The *Metadatabase Manager* coordinates activity occurring in the *metadata*base which, it should be remembered, includes all information and knowledge about the system. Upon request from the *User*, navigation through the *metadata*base is controlled by the *Metadatabase Manager*, and is made more efficient by using user profile and views information. The *Query Processor* module intelligently decomposes and translates *User* queries, generating information requests to the science database. The *Database Manager* performs the functions of the typical database, performing search procedures when queries are received. The *Meta/Science Data Update System* intelligently extracts metainformation from incoming science data and determines initial links (i.e., potential navigation paths) and associations among the new and already existing data. As a result, the existing knowledge structure in the *metadata*base is modified to include the new information. Original science data, now with appropriate *metadata* linking information attached, is then stored in the science database.

A second major part of Phase 1 prototype development, is that part of the implementation which focuses on designing standardized message passing protocols and data structures, allowing users operating any hardware or software to interact with the IIMS as well as allowing external DBs to update IIMS information in a consistent and standard fashion. Several approaches to developing automated data information update capability in the *Meta/Science Data Update System* are possible. The conventional approach would be to first define all existing and projected data and then build a knowledge structure based on the contents and characteristics of those data. The result would be a knowledge structure analogous to a card catalog in a library. Such an approach presents several long-term problems: automatic information updates and general knowledge structure operations would be possible only within the bounds of the predefined data. As a user develops new ways to use existing instruments or analyze raw data, or when new instruments are deployed, new types of data will be produced. Modifications to the existing knowledge structure would have to be manually implemented. Also, it is highly unlikely that PIs will have the time or inclination to research and cross reference characteristics of new data to those of seemingly unrelated disciplines or problems. The capability and effective usefulness of the *Meta/Science Data Update System* would be limited to the contextual boundaries set by the user based on personal knowledge. This approach would likely provide an acceptable solution for the near-term, but it does not address long-term requirements. Standard data definition approaches do not provide the information needed for this level of intelligent operation.

To provide the types of information to make this level of capability possible, new techniques for describing data are needed. It is important to understand that to implement such a capability, not only data contents (traditionally described in a Data Description Language), but data characteristics, in a form suitable for cross referencing to other data types, must

be addressed. One possible solution would be a structure which we term a *Data Set Descriptor* (DSD). A properly developed DSD would describe data in such a way that relational cross references can be made to characteristics of other data types. This would also give the *Meta/Science Data Update System* the capability of automatically creating new data categories and adjusting the knowledge structure based solely on the DSD input for a new data type. By having properly defined characteristics of this new data set, the *Meta/Science Data Update System* would be able to determine if the new data fit the existing knowledge structure or if the knowledge structure should be modified. The *Meta/Science Data Update System* would also be able to establish inferences required to recognize that new data may be useful for other, peripherally related investigations. Implementation of a DSD type data structure provides the foundation necessary for smooth system evolution. As new advances are made in AI and data representation techniques, the DSD provides the basis for integrating advances without massive system restructuring. An additional advantage to this approach would be the virtual elimination of software changes in the system to maintain data compatibility.

4.6.2. Phases 2-4

Primary modifications to the system developed during Phase 1 include the implementation of a true 'remote' user capability, the implementation of user communications control in a separate piece of hardware (Zenith 100[5]), the implementation of an external DBMS on a separate piece of hardware (the second microExplorer), and initial enhancements of several skeleton functions initially developed during Phase 1 (e.g., adding more support features in the intelligent interface module, adding more advanced query processing capability). While some functional enhancements will be added to the system, the primary objective during Phase 2 is to distribute functionality of the system developed during Phase 1 to other networked hardware components. The primary objective during Phase 3 is to significantly enhance the capabilities of both the *Intelligent Interface* module, the *Meta/Science Data Update System*, and the *Query Processor*, in addition to expanding the number and types of potential DBMSs with which the IIMS might have to interface (including the incorporation of an existing science database). Anticipated enhancements include the capability to handle multiple users with text or graphics-oriented user-interfaces, and the addition of an automated planning and scheduling capability that will allow resolution of user data request conflicts. Currently, it is anticipated that enhancements to the system during Phase 4 will include interfacing with X WINDOWS, increasing the number of distributed databases, providing on-line user data path switching, and simulating data capture, allowing for testing of alternative IIMS processing and storage topologies. To date, our work has focused on several IIMS areas and initial results of our prototyping work are briefly described in section 5.

[5]The Zenith 100 has unique communications capability with 32 available ports.

5. RESULTS AND CONCLUSIONS

IIMS Operations Concept. We have completed a detailed examination of potential operations and associated functional requirements for a Space Station-era IIMS. This analysis served as the focal point for our prototyping work.

Data Modeling. We have been actively pursuing information concerning potential types of scientific data with which the advanced IIMS will have to be familiar. Discussions we have had with National Center for Atmospheric Research personnel, as well as documentation concerning typical satellite data related to atmospheric studies and sample data they have provided, have allowed us to more accurately define potential types of data and queries used to access that data. The sample data we have obtained includes data at a minimum of two processing levels and will include data that may be displayed in a graphical format. It is our intention to use the sample data, not only to provide insight into real scientific data structures but also as drivers for graphic displays that will likely be part of the IIMS's functional repertoire (e.g., browse data, quick-look data). We have also met with Earth Resources Observation System (EROS) data center personnel to discuss their data storage, access, handling, and distribution requirements and have received documentation concerning the types of data and database technology currently in place at the EROS data center. Information gathered from these discussions has helped focus prototype activity in several areas (e.g., user-interface requirements, accessing protocols, distributed database management and handling).

We have begun design of the module that will handle update messages received by the IIMS from heterogeneous data sources, and analysis of potential technical difficulties is proceeding. One issue currently being addressed is the format of update messages. Because of differences in types of scientific data (e.g., image, text, numerical) and their corresponding attributes, a variable format system is being analyzed for suitability to this environment. Problems now being studied include how to allow (1) automatic generation of these variable update messages, and (2) interactive generation of these messages in which the software handles proper message formatting depending, in addition to several other factors, upon the data type. We have also begun to analyze how *metadata* should be stored in the IIMS database, and the manner in which intelligent support features can be incorporated to allow efficient access to data. Technical issues involve both the representation of the knowledge and its automatic generation and incorporation into the system. Initial results indicate an object-oriented data representation technique to be the most flexible given projected functional

requirements the *metadata* must support.

User-system Interface. An analysis of NASA's Space Station Information System human-computer interface requirements has led to the initial prototyping of one potential IIMS user-interface concept. The user-interface is interactive and dynamic as well as both graphic and text oriented, allowing for multiple screen configurations suited to different IIMS functions.

Catalog Management. The *Data Management System* (DMS) *Tool* has been developed to provide data management capabilities for the TI microExplorer. It is based on the relational data model with a few extensions to allow for more flexibility. The system provides both a *Data Definition Language* (DDL) and a *Data Manipulation Language* (DML). The DDL allows for the definition, destruction, saving, loading and structural modification (adding, deleting and modifying of relational attributes) of databases and relations. The DML provides for an unrestricted query capability and relational set operations (join, intersection, union and difference). The query capability allows the user to provide any s-expression (LISP expression) as the selection criteria and project any combination of attributes from the result. The system provides for simple concurrency control through a series of locks and tokens on the database and relational levels (single modifier, multiple concurrent access). Multiple databases can be active at one time; however, querying between databases is not yet supported. Examples of DBMS functions that are not currently implemented are query optimization, transaction processing, index capability, crash recovery, and other advanced DBMS capabilities. The DMS will initially be used to implement the IIMS database and handle other IIMS prototype database needs. It has already been used to demonstrate how dynamic menus of IIMS database items can be generated from an IIMS database.

Interactive Graphics Objects (IGO). This system was developed to provide for dynamic binding of interactive command structures to mouse-sensitive graphical objects. The user can build command tables consisting of actions that result from mouse button activation; documentation for the appropriate activity is displayed whenever the mouse is over the object. Various combinations of commands can become "command modes" which can then be bound to different graphical objects; these IGOs can be manipulated just like any other graphics object on the microExplorer, and when the mouse is positioned over one of these objects, the specified "command mode" will become active. IGOs can be used to provide a wide range of capabilities. Because of the general capability which this system provides, it has been used to implement an interactive graphical display of the catalog hierarchy and an initial interactive tutorial system that simulates IIMS functionality (including animation and interactive *help*).

Communications Protocol and Message Passing System. An initial message passing/communications protocol format has been established and will be tested this year in terms of remote user access and automated data updates to the IIMS. Instrumental in the development of this protocol was the the implementation of a communications program providing direct user communications and querying capability between any combination of nodes on the MOL AppleTalk network. Also completed were network monitor and control programs. Current development efforts are focused on understanding and using the Macintosh RPC protocol. To date, we have successfully implemented a low level data transfer function which consists of each system requesting and receiving the current system time from the other. We are currently working to expand the function's capability and provide interfaces to Macintosh system applications.

Query and Browse Facility. Several topics related to IMS-querying capability have been studied and progress has been made on prototype functional implementation in some of these areas. Topics of focus have included (1) types of queries which will need to be handled by the IIMS, (2) identifying and prototyping special requirements necessary for handling spatial queries, (3) query formulation, and (4) the structure of the associated knowledge base(s). We will use several techniques to aid the IIMS-user in the location of information. Some of the techniques we have begun to analyze are constrained query formulation (including spatial and temporal selection), conceptual views (aided by user profiles), browsing, knowledge base assistance, cooperative response, query reformulation, suggestions of related data, and various forms of on-line *help*. A second issue we have begun to study is that of integrating browsing and querying facilities for the IIMS. This work is in the early stages of analysis, but we currently expect that browsing will probably provide the user with the fastest method for dynamically limiting the scope of the data for a particular query, for which no view is defined. As the user browses through the data hierarchy, the system will place restrictions on any query formed. Much work needs to be done on the definition of a database structure that will allow these two methods to be smoothly integrated. Current prototype implementation includes a constrained query formulation capability along with allowing the user to graphically select spatial information concerning data of interest. The user can browse IIMS *metadata* through either a menu or graphical representation.

Tutorial and User profile Capabilities. Both of these functions are in the rudimentary stage with a basic user profile capability being established. Currently, user profiles may be modified through either an IIMS operator's screen or through the user's own screen. Since objects are used as the representation technique for users, modifications to profiles are easily made. Future work will begin to tie user profiles to querying and browsing capabilities. Extensive use of the

IGO system will be made to develop the prototype on-line tutorial system.

Work described in this paper suggests several conclusions: First, concerning our approach, use of rapid prototyping and AI programming environments appears to be an efficient and cost-effective means of testing several proposed requirements for projected advanced ground systems. Second, our work in advanced information management and access has given us the springboard to test more sophisticated IIMS concepts in a highly modular and dynamic environment. The working prototype we have developed has already demonstrated the feasibility of applying object-oriented programming and advanced database concepts to problems associated with projected Space Station-era information management systems. Future work will continue to extend the current IIMS prototype by testing several advanced IIMS features, particularly in the area of remote, distributed access.

REFERENCES

1. Blaha, M. R., Premerlani, W. J., & Rumbaugh, J. E. (1988). *Relational Database Design Using an Object-oriented Methodology.* Communications of the ACM, 31(4), 414-427.

2. Campbell, W. J., Roelofs, L. H., & Short, N. M. (1987). *The Development of a Prototype Intelligent User Interface Subsystem for NASA's Scientific Database Systems.* NASA Technical Memorandum 87821.

3. Carnahan, R. S., & Mendler, A. P. (1987). *Rapid Prototyping and AI Programming Environments Applied to Payload Modeling.* Proceedings of the Third Conference on Artificial Intelligence for Space Applications, Huntsville, Alabama, 255-260.

4. Conklin, J. (1987). *Hypertext: An Introduction and Survey.* In I. Greif (Ed.), Computer-Supported Cooperative Work: A Book of Readings. Morgan Kaufmann, 423-475.

5. Hutchins, E. L., Hollan, J. D., & Norman, D. A. (1986). *Direct Manipulation Interfaces.* In D. A. Norman, & S. W. Draper (Eds.), User Centered System Design. Lawrence Erlbaum Associates, 87-124.

6. King, R. (1986). *A Database Management System Based on an Object-oriented Model.* In L. Kerschberg (Ed.), Expert Database Systems. Benjamin/Cummings, 443-468.

7. Li, Q., & McLeod, D. (1988). *Object Flavor Evolution Through Learning in an Object-oriented Database System.* Proceedings of the Second International Conference on Expert Database Systems, Tysons Corner, Virginia, 241-256.

8. McArthur, D. J., Klahr, Pl, & Narain, S. (1986). *ROSS: An Object-oriented Language for Constructing Simulations.* In P. Klahr, & D. Waterman (Eds.), Expert Systems: Techniques, Tools, and Applications. Addison-Wesley, 70-91.

9. NASA (1988). *A Program For Global Change - Earth System Science: A Closer View.* A Report of the Earth System Sciences Committee NASA Advisory Council.

10. NASA Goddard Space Flight Center (1986). *Earth Observing System Data and Information System Report of the Eos Data Panel - Volume IIa.* NASA Technical Memorandum 87777.

11. NASA Goddard Space Flight Center (1988). *Customer Data and Operations System (CDOS) Concept Definition Document.* NASA GSFC.

12. Norman, D. A. (1986). *Cognitive Engineering.* In D. A. Norman, & S. W. Draper (Eds.), User Centered System Design. Lawrence Erlbaum Associates, 31-61.

13. Polson, M. C., & Richardson, J. J. (Eds.) (1986). *Intelligent Tutoring Systems.* Lawrence Erlbaum Associates.

14. Short Jr, N. M., Campbell, W. J., Roelofs, L. H., & Wattawa, S. L. (1987). *The Crustal Dynamics Intelligent User Interface Anthology.* NASA Technical Memorandum 100693.

15. Short Jr, N. M., Wattawa, S. L. (1988). *The Second Generation Intelligent User Interface For The Crustal Dynamics Data Information System.* Proceedings of the 1988 Goddard Conference On Space Applications Of Artificial Intelligence, Greenbelt, Maryland, 313-327.

16. Stefik, M., & Bobrow, D. G. (1986). *Object-oriented Programming: Themes and Variations.* AI Magazine, 6(4), 40-62.

17. Wenger, E. (1988). *Artificial Intelligence and Tutoring Systems.* Morgan Kaufmann.

18. Zhu, J., & Maier, D. (1988). *Abstract Objects in an Object-oriented Data Model.* Proceedings of the Second International Conference on Expert Database Systems, Tysons Corner, Virginia, 3-16.